NUI Galway
OÉ Gaillimh

Semester 2 Examinations 2015/2016

Exam Code(s)	4BCT, 4ECE
Exam(s)	B.Sc. in Comp Sc. & Information Technology B.E. in Elect.& Comp. Engineering
Module Code(s)	CT420
Module(s)	Real-Time Systems
Paper No.	1
Repeat Paper	N
External Examiner(s)	Dr. John Power
Internal Examiner(s)	Prof. Gerard Lyons Dr. Michael Madden *Dr. Hugh Melvin *Dr. Michael Schukat

Instructions: Answer Q1 and 1 other Q from section A.
Answer 2 questions from section B.
All questions carry equal marks.

Duration	2 hours
No. of Pages	7
Discipline	Information Technology

Requirements:
MCQ
Handout
Statistical/ Log Tables
Cambridge Tables
Graph Paper
Log Graph Paper
Other Materials

Release to Library: Yes [X] No [ ]

Section A

Q1

(i) Distinguish between time, phase and timing synchronisation, using examples from the power systems domain to illustrate your answer. [10]

(ii) Quality of Experience (QoE) is a related though much broader concept than Quality of Service (QoS) . Distinguish between the two terms, using the MMOG (Massively Multiplayer Online Gaming) application as an example.

[6]

(iii) Distinguish between narrowband (NB), wideband (WB), and super-wideband (SWB) voice codecs

[8]

(iv) You have access to a network capture file of a WebRTC session between two participants. Explain what you would expect to see, focusing on the following:

- Call set up
- Media Transport
- Lip-synch implementation [16]

Q2

(i) Distinguish briefly between intramedia, intermedia, multi-destination , and multi-source intermedia synchronisation, using examples to illustrate your answer. Outline also which of the following are main concern(s) – time/phase/timing synch, for each of the four types of media synch .

[12]

(ii) The data shown below outlines the output of the **ntpq** utility from 2 NTP clients- A and B. Based on the output, it is clear that client B has better synch to UTC. Explain in some detail why this is the case - your answer should comment on redundancy, refID, delay, offsets, jitter, stratum level, and reachability.

Client A

remote	refid	st	t	when	poll	reach	delay	offset	jitter
+196.49.0.37	212.47.239.163	3	u	812	1024	356	104.182	4.781	1.807
+zibbi.meraka.cs	0.60.139.194	2	u	394	1024	253	234.859	-8.110	0.287
*golem.canonical	192.150.70.56	2	u	151	1024	377	22.811	1.683	0.744

Client B

remote	refid	st	t	when	poll	reach	delay	offset	jitter
*140.203.16.5	.GPS.	1	u	91	1024	377	0.322	-0.442	0.017
-140.203.204.77	.GPS.	1	u	935	1024	377	1.804	0.167	0.854
-ntp-sop.inria.f	.GPS.	1	u	323	1024	377	39.380	-1.899	0.049
+ntp1.rrze.uni-e	.DCFp.	1	u	340	1024	377	41.440	-0.445	0.050
+ntp2.ja.net	193.62.22.74	2	u	329	1024	377	9.012	-0.466	0.192
-ntps1-0.eecsit.	.PPS.	1	u	393	1024	220	34.614	-0.350	0.650
-192.33.214.47	129.194.21.195	2	u	386	1024	253	28.113	-0.532	0.062

[18]

(iii) The Precision Time Protocol (PTP) is used in many utilities to deliver usec level time synchronisation. It does this through **hardware level timestamping** and **on-path support**. Explain briefly what these two terms means, and why they are important for the precise operation of PTP. [10]

Q3

(i) Mouth to Ear delay is a key metric for Voice over IP. Explain what it is, why it is a key metric, and outline the main contributory elements on the sender, network and receiver.

[20]

(ii) Distinguish between <u>Media Independent</u> and <u>Media Dependent</u> Forward Error Correction (FEC) strategies for Voice over IP, using examples to illustrate your answer. Comment also on their respective impact on delay, bandwidth utilisation, and quality

[10]

(iii) MPEG-DASH is very widely used for streaming multimedia. Explain briefly what it is, how it works and why it works well in today's best effort Internet.

[10]

Section B

Q4

(i) Briefly distinguish between (a) deeply embedded systems (ES), (b) shallow ES, (c) medium ES, (d) commodity ES and (e) critical ES. Provide examples <u>for any 4</u> of these categories.

[6]

(ii) Memory locking is an important feature of POSIX 4. Explain the code snippet below and use the 7-state process model to outline why memory locking is important for (hard) real-time systems.

```
/*Main routine */
int main(void ){
/* Lock all process down */

mlockall(MCL_CURRENT|MCL_FUTURE);

… process code

munlockall();
return 0;
}
```

[10]

(ii) Using an example explain how a stack overflow can be caused and how it compromises the run-time stack of a process.

[12]

(iii) Summarise the inner workings of the Hamming (7, 4) code, and, using examples, showcase its error correcting and error detecting capabilities.

[12]

Section continues on next page.

Q5

(i) Construct a cyclic executive (CE) schedule for the set of tasks listed in Table 1 below. Further-on, draw the CE timeline and determine how much CPU time is left for the processing of asynchronous interrupts.

Task	Period p msec	Exec Time msec
A	25	10
B	25	5
C	50	5
D	50	5
E	100	2

Table 1

[12]

(i) Distinguish between *static software redundancy* and *dynamic software redundancy*. What are the advantages and disadvantages / problems of both approaches? [6]

(ii) Distinguish between RAID-0, RAID-1 and RAID-5.
Further on show how (a) a 2-disks RAID-0, (b) a 2-disks RAID-1, and (c) a 5-disks RAID-5 system respond to the failure of an entire hard disk.

[10]

(iii) Using an example, show how graph-based bound calculations can be used to determine the worst-case execution time (WCET) of program code. Your example must include one loop and two if-then-else statements.

[12]

Section continues on next page.

Q6

(i) Using the task set in Table 2 below:
- Calculate the overall *CPU utilisation U*.
- Determine the process schedule using the *RM scheduling algorithm*.
- Determine the process schedule using the *EDF scheduling algorithm*.
- Comment on the difference between both schedules.

Task	Execution Time	Period
1	10	50
2	20	100
3	30	150
4	50	200

Table 2

[14]

[10]

(ii) What is meant by the *priority inversion problem* and how can it be solved? Use an example to illustrate your answer.

[10]

(iii) Summarise the main features of POSIX signals. Further on, outline the purpose and functionality of the structure *sigaction* below and show how this structure allows the handling of queued signals as defined in POSIX.4.

```
struct sigaction{
   void (*sa_handler)();
   sigset_t sa_mask;
   int sa_flags;
   void(*sa_sigaction)(int, siginfo_t *, void *);
};
```

[10]

(iv) Distinguish between the fault tolerant techniques (a) hardware redundancy, (b) graceful degradation and (c) fail-soft. Using examples show how these concepts are adopted in an automotive context, e.g. where they can be found in cars.

[6]